Journal of Computational and Applied Mathematics 28 (1989) 119–127
North-Holland

A method of parallel iteration

J.M. BARRY *
Department of Mathematics, University of Tennessee, Knoxville, Tennessee 37796, U.S.A.

J.P. POLLARD
Australian Nuclear Science and Technology Organisation, Private Mail Bag 1, Menai NSW 2234, Australia

E.L. WACHSPRESS
Department of Mathematics, University of Tennessee, Knoxville, Tennessee 37796, U.S.A.

Received 9 May 1988
Revised 23 November 1988

Abstract: A method of parallel iteration for large systems of linear equations based on coarse mesh rebalance procedures is described. Multiplicative and additive forms of correction are applied and tested. Finally a hybrid correction stencil is introduced. Some matrix properties of intermediate linear systems are considered.

Keywords: Parallel iteration, coarse mesh rebalancing, multiplicative and additive correction.

1. Introduction

In reactor physics computations, it is commonplace to accelerate the rate of convergence of iterative schemes by techniques based on variational principles [7]. These techniques frequently are referred to as coarse mesh rebalance (CMR) methods. They are the fore-runners of the multigrid method [3], although only two grid levels are used. Convergence is accelerated by CMR through the reduction or removal of low frequency error components [6]. These highly effective CMR techniques seek an approximation to the original problem over a much coarser grid than that used in the fine grid idealization of the physical continuum. When the coarse grid problem is solved, the current estimate for the fine grid solution is adjusted by an appropriate correction mechanism. One of two forms of adjustment, multiplicative or additive correction is commonly applied. The coarse grid problem is defined generally on a much coarser scale than that of the original problem, so it is frequently solved by direct rather than iterative schemes. The properties of the coarse mesh matrix are determined by the variational procedures used for its generation and in some cases, the well-known iterative methods are not appropriate. In this work we consider the simplest additive and multiplicative forms of correction arising from coarse mesh experience, as the basis of an algorithm suitable for parallel architectures. Unlike some CMR

* Present address: Australian Nuclear Science and Technology Organisation, Private Mail Bag 1, Menai NSW 2234, Australia.

approaches, the intermediate systems of equations that are generated are sufficiently large to warrant iterative schemes of solution. Finally, a hybrid scheme of correction is introduced to overcome the difficulties that may arise when the simplest form of additive correction is employed.

2. Coarse mesh rebalance procedures

Consider a large system of linear equations

$$Ax = b, \tag{2.1}$$

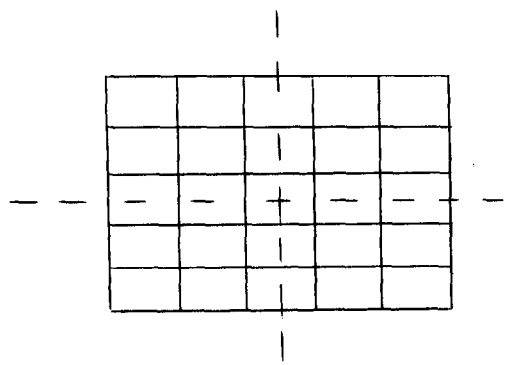
obtained after a suitable discretization has been applied to a partial differential equation of the form

$$-\nabla \cdot D\ \nabla\phi + \sigma_{\text{rem}}\phi = S, \tag{2.2}$$

subject to suitable boundary conditions. In reactor physics, for the case of a single energy group, ϕ is the neutron flux, D is the diffusion coefficient, σ_{rem} the removal cross section and S the neutron sources lumped into a single term. The solution ϕ is always positive in reactor physics computations and in this paper we restrict our attention to this situation, however, our techniques are suitable for more general applications. The $n \times n$-matrix A of (2.1) is real, sparse, symmetric and positive definite.

In forming an $m \times m$-coarse mesh approximation to (2.1), we are guided by geometric considerations. For example, in a two-dimensional problem we could consider the fine mesh on which (2.2) is idealized to be overlayed with a coarse grid (Fig. 1). For the simple form of CMR as is used here, we obtain a single correction factor for each coarse mesh region. The solution at all grid points of the fine mesh covered by that coarse mesh segment are adjusted with it. The correction factor may be used to compute a multiplicative correction

$$x^{(t+1)} = Cx^{(t)} \tag{2.3}$$



$$P^T = \begin{bmatrix}
1&1&1&0&0&0&1&1&1&0&0&0&1&1&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&0&0&1&1&1&0&0&0&1&1&1&0&0&0&1&1&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&1&1&1&0&0&0&1&1&1&0&0&0&1&1&1&0\\
0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&1&1&1&0&0&0&1&1&1&0&0&0&1
\end{bmatrix}$$

Fig. 1. Partitioning of a 6×6-fine grid into a 2×2-coarse mesh, and the corresponding matrix P.

to the previous iterate $x^{(t)}$. C is a diagonal matrix whose n nonzero elements are composed of m distinct values corresponding to the multiplicative correction factors for each coarse mesh region. The diagonal of C is given by

$$\text{diag}(C) = Pc_{(\text{mult})}.$$

Alternatively an additive correction

$$x^{(t+1)} = x^{(t)} + Pc_{(\text{add})}, \tag{2.4}$$

may be used. In each of the above, P is an $n \times m$-selection matrix, while the vector c of length m contains the individual correction factors for each respective coarse mesh region imposed over the fine grid. The matrix P has elements 0 or 1, and its pattern is determined by the geometrical association of the correction factors and the matching coarse mesh regions. A typical example is shown in Fig. 1. Perhaps the simplest coarse mesh approximation for determining the multiplicative correction factors of (2.3) is given by applying the weighted residual method to (2.3) with simple P^T weighting. The notation X is introduced to denote a diagonal matrix whose nonzero terms are the elements of $x^{(t)}$. For the neutron diffusion problem every element of $x^{(t)}$ is strictly positive. The CMR approximation is

$$(P^TAXP)c_{(\text{mult})} = P^Tb. \tag{2.5}$$

The $m \times m$-matrix (P^TAXP) is not symmetric, row diagonal dominance is lost, although column dominance is preserved. Iterative schemes that rely on symmetry are not suitable for solution of equation (2.5) and in many nuclear codes as $m \ll n$ direct solution techniques are used. (The efficient method of implicit nonstationary iteration MINI [1] is not so restricted by symmetry and is a suitable technique for this situation.)

If Galerkin weighting were applied instead in the residual method, the $m \times m$-coarse mesh system of equations

$$(P^TXAXP)c_{(\text{mult})} = P^TXb \tag{2.6}$$

is produced. The matrix (P^TXAXP) has the same properties as that of the original matrix A [6], so iterative techniques such as the conjugate gradient method are applicable. For additive correction, the Galerkin weighted residual technique gives the symmetric system

$$P^TAPc_{(\text{add})} = P^T(b - Ax^{(t)}). \tag{2.7}$$

It is easy to demonstrate the properties of the original matrix are preserved in (P^TAP). In this work we will consider the Galerkin weighted systems (2.6) and (2.7) as choices for the driving mechanism for our parallel algorithm.

3. Parallel algorithm

Consider the partial differential equation (2.2) defined over a three-dimensional physical domain where a seven-point finite-difference approximation is applied over a rectangular grid of size $N_x \times N_y \times N_z$. Further, if there are K processors at our disposal, the grid can be divided between them evenly into K segments. Each segment consists of N_z/K (x, y)-planes of grid points. This orientation of planes is suggested by the nature of the reactor physics problem as

there is generally more detail in the mesh and more material changes across the reactor, because coolant channels, fuel rods, etc., are directed in the third spatial dimension.

It would seem desirable in a parallel approach for each processor to handle a different segment of the grid in full geometric detail, while knowledge of the remaining segments is included but in progressively less detail as we move further away from the fully detailed segment. Coarse mesh rebalance procedures provide an easy method of achieving this aim as grid points not required in full mesh detail are readily "collapsed" away from the fully detailed segment. Individual correction factors are sought at each gridpoint for the segment held in fine mesh representation and for each coarse mesh node produced by "collapsing". The solution of the K newly formed subsystems constitutes one global iteration of the parallel algorithm. Correction factors only from those planes solved in fine detail are used to correct the estimate for the next global iteration. The remaining factors are simply discarded. The overheads in computing the discarded factors are acceptable. The introduction of coarse nodes outside the fine mesh segment is based on a halving rule—for the first planes adjacent to the kth fine mesh segment $\frac{1}{4}(N_x \times N_y)$ coarse correction factors are required, then $\frac{1}{16}(N_x \times N_y)$ and so on. Consequently for each processor, the bulk of the computation will be spent in determining the mesh correction factors for the grid points held in fine detail. Because the multiplicative (2.6) and additive (2.7) systems are of a sufficiently high order and as a reasonable solution estimate is usually available only iterative methods of solution are considered. The K coarse mesh systems generated are solved in this work with an incomplete Cholesky conjugate gradient scheme.

Before further examination of the parallel approach is considered, it is interesting to consider several aspects of computing the correction factors when no "collapsing" of grid points occurs and to compare such a calculation with the solution of the original system (2.1). That is, instead of solving (2.1) for neutron flux we solve (2.5), (2.6) or (2.7) with $m = n$. For $m = n$, $P = I$ (the identity matrix) and the additive form (2.7) becomes

$$A\boldsymbol{c}_{(\text{add})} = b - A\boldsymbol{x}^{(0)}, \tag{3.1}$$

while the multiplicative forms (2.5) and (2.6) become

$$AX\boldsymbol{c}_{(\text{mult})} = \boldsymbol{b} \tag{3.2}$$

and

$$XAX\boldsymbol{c}_{(\text{mult})} = X\boldsymbol{b}, \tag{3.3}$$

respectively. The diagonal matrix X in this situation has $\boldsymbol{x}^{(0)}$ for its nonzero components. Examination of the additive form (3.1) shows the iterative matrix is the same as that used for the original system, consequently the same level of computational effort as is used to solve the original system, is required, no matter which iterative technique is selected. Although for reasons of lack of symmetry, (3.2) is not considered in the parallel algorithm, its solution is the same as that of the Galerkin weighted form (3.3). (For $n \neq m$ this situation, however, does not apply.) In addition, should the trial solution $\boldsymbol{x}^{(0)}$ be a unit vector, both multiplicative forms are identical to the original system. For other trial solutions, the matrices arising in the multiplicative forms differ from the original. For some iterative schemes, however, the pattern of convergence is the same as that of the original system, provided $\boldsymbol{c}^{(0)}_{(\text{mult})} = 1$ (the unit vector) is used to start the iterative process.

It is easy to show that the solution of (3.2) and (3.3) by the Gauss–Seidel methods, for this special instance of $m = n$, gives the same convergence pattern as that of the original system. The

matrix A can be expressed $A = E + B + E^{\mathrm{T}}$, where E is lower triangular, and B is diagonal. Suppose $\boldsymbol{x}^{(0)}$ is some initial approximation to the solution of the original system, and $\boldsymbol{c}^{(0)}_{(\mathrm{mult})} = 1$ is the corresponding initial trial value in the iterative solution of (3.3). For the Galerkin weighted multiplicative correction scheme

$$\boldsymbol{c}^{(1)}_{(\mathrm{mult})} = \left[X(E+B)X\right]^{-1}\left[Xb - XE^{\mathrm{T}}X\boldsymbol{c}^{(0)}_{(\mathrm{mult})}\right],$$

which on multiplication by X and simplification yields

$$\boldsymbol{x}^{(1)} = (E+B)^{-1}(b - E^{\mathrm{T}}\boldsymbol{x}^{(0)}),$$

which is the first Gauss–Seidel iterative step for (2.1). The proof is completed by induction. The proof is similar for the non-Galerkin form (3.2).

In the case of the seven-point finite-difference scheme used here, the convergence pattern for the incomplete Cholesky conjugate gradient scheme with no infill [5] is the same for the original scheme and (3.3). The incomplete Cholesky decomposition for the matrix A with no infill for this differential operator is (LDL^{T}), where the lower triangular elements of L are the same as the corresponding elements of A. It can be shown that the incomplete Cholesky decomposition for the Galerkin form XAX of (3.3) is $(XLDL^{\mathrm{T}}X)$. Given this, the solutions of (2.1) or (3.3) by incomplete Cholesky decomposition are equivalent. This is apparent by comparing the corresponding steps of the preconditioned conjugate gradient algorithm. For example, consider the incomplete Cholesky solution step for calculating z_{t-1} (following [4]) which for the case of the multiplicative form (3.3) is

$$\boldsymbol{z}_{t-1} = (XLDL^{\mathrm{T}}X)^{-1}\left(X\boldsymbol{b} - XAX\boldsymbol{c}^{(t-1)}_{(\mathrm{mult})}\right) = X^{-1}(LDL^{\mathrm{T}})^{-1}(b - A\boldsymbol{x}_{t-1}).$$

Computation and simplification of the numerator in β_t of the next step gives

$$\langle \boldsymbol{z}^{\mathrm{T}}_{t-1}, \boldsymbol{r}_{t-1}\rangle = \langle (LDL^{\mathrm{T}})^{-1}(b - A\boldsymbol{x}_{t-1}), (b - A\boldsymbol{x}_{t-1})\rangle,$$

which is immediately identified as the equivalent numerator in β_t of Golub and Van Loan [4] for the preconditioned conjugate gradient solution of the original system (2.1).

Without incomplete Cholesky preconditioning, it may be verified that the conjugate gradient scheme does not lead to the same pattern of convergence for the two schemes ((2.1) and (3.3)). Limited numerical experimentation suggests that without preconditioning the conjugate gradient solution of the original system is faster than the solution of the multiplicative correction form.

The implicit scheme MINI, of interest to two of the authors is written

$$\boldsymbol{x}^{(t)} = (E + B + \Gamma)^{-1}\left(\boldsymbol{b} - (E^{\mathrm{T}} - \Gamma)\boldsymbol{x}^{(t-1)}\right),$$

where Γ is a diagonal matrix whose elements are $\Sigma_{j>i} a_{ij}\gamma^{(t-1)}_{ij}$. The values of $\gamma^{(t-1)}_{ij}$ are non-stationary and are bounded $(0 \leqslant \gamma^{(t-1)}_{ij} < \min(1, x_j/x_i))$. The rules for calculating $\gamma^{(t-1)}_{ij}$ [1] although not specified here are sufficient to ensure the two schemes are not equivalent. Limited numerical experimentation marginally favours the multiplicative form over the original.

4. Results for additive and multiplicative correction schemes

The usefulness of the CMR parallel approach has been demonstrated [2] for multiplicative correction forms. Several enhancements not considered here are appropriate for improving its

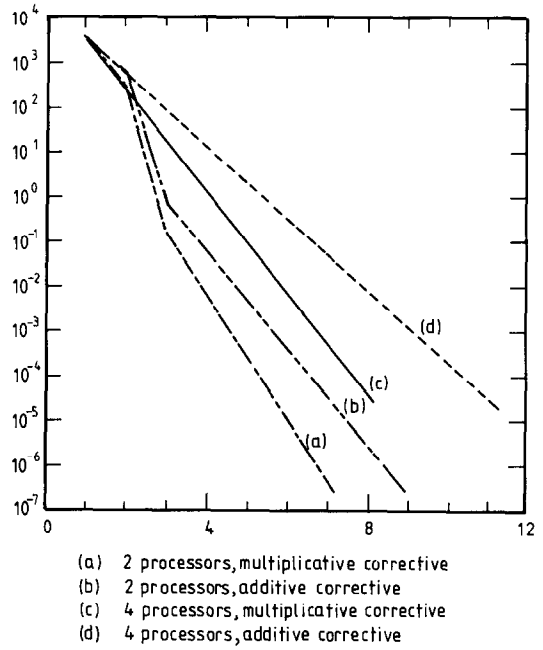


Fig. 2. Performance of multiplicative and additive schemes on trial problem.

performance. In this work we are content to compare the additive and multiplicative forms. Both were tested on a $32 \times 32 \times 32$ rectangular discretization of the partial differential equation

$$-\nabla \cdot D\ \nabla\phi = S,$$

defined with $\phi = 0$ on the boundary, of a cube of side length π. This and other trial problems are described in more detail in [2]. The diffusion coefficients are 1.0, 2.0, 3.0 and 4.0 and they apply to 16×16 subgrids for every (x, y)-plane. The subgrids are ordered clockwise from the top left corner. The source term S is computed numerically for the solution $\phi = \sin x \sin y \sin z$, while $\phi = 0.5$ is used to start the parallel iterative scheme. The grid points are equally distributed in the z-direction, while within each plane they are distributed such that the grid incremental values are proportional to $\delta x_i = 2 - \cos(\frac{1}{33} i\pi)$. To compare the effectiveness for global convergence of the multiplicative and additive forms (2.6) and (2.7) the decay in error $\| \phi - \phi^{(t)} \|$ is plotted as a function of the number of global iterations in Fig. 2 for the case of two and four processors; while results for two, four, eight and sixteen processors are tabulated in Table 1. The process is terminated when four-figure accuracy is obtained at all grid points. To keep the comparison of the two methods in step, all the intermediate solutions are computed to five-figure accuracy. (Normally loose convergence would be accepted at the early stages.) Although results for only one trial problem are presented here the findings are indicative of wider testing.

The multiplicative scheme decays more rapidly at the early stages. In the iterations the shapes of the two error decay terms are more similar, the multiplicative, however, is slightly steeper. The computational effort in generating the additive form is less than with the multiplicative scheme, because the left-hand side matrix is simpler and does not alter for subsequent global iterations, however, the multiplicative form appears superior. It is postulated, that the difficulty with the

Table 1
Global iteration count for additive and multiplicative correction of trial problem

Number of processors	Additive correction	Multiplicative correction
2	9	7
4	11	8
8	22	19
16	31	28

additive form arises because the additive correction technique is not as capable of correcting errors over coarse grid sections, particularly those near the boundary when the intermediate solutions are not flat. In this problem the trial solution is flat, so the two methods are the same for the first global iteration. After this iteration the intermediate solution is no longer flat. Consequently it is more difficult to obtain a useful single additive correction for the next iteration at the coarse mesh points. When the shape of the trial solution becomes similar to that of the real solution this effect is diminished greatly. Because the additive corrections were not to be applied to the coarse grid points in the parallel algorithm, it was hoped at the start that the additive form of correction would be adequate; practical experience indicates otherwise. The multiplicative correction scheme on the other hand is capable of rescaling (and hence changing the shape) of the intermediate solutions across the coarse grid. This allows a reduction in the overall error and consequently gives some computational advantage to the multiplicative form.

5. Hybrid correction scheme

In an effort to overcome some of the difficulty with the additive scheme during early iterations, the authors have experimented with a hybrid correction scheme. In this, additive correction is applied to grid points maintained in fine mesh representation but multiplicative correction is applied elsewhere. Essentially in the hybrid scheme, only the coarse mesh section of the resulting matrix needs to be recomputed on each global iteration. The hybrid correction scheme is expressed

$$\boldsymbol{x}^{(t+1)} = \sum_{k=1}^{K} c_{(\mathrm{mult})_k} P_{(\mathrm{mult})_k} \boldsymbol{x}^{(t)} + \left[\sum_{k'=1}^{K'} P_{(\mathrm{add})_{k'}} \boldsymbol{x}^{(t)} + \sum_{k'=1}^{K'} c_{(\mathrm{add})_{k'}} P_{(\mathrm{add})_{k'}} 1 \right],$$

where the $P_{(\mathrm{mult})}$ and $P_{(\mathrm{add})}$ are diagonal partitioning matrices of order n used to extract the appropriate grid points for correction. $\Sigma_{k=1}^{K} P_{(\mathrm{mult})_k} + \Sigma_{k'=1}^{K'} P_{(\mathrm{add})_{k'}} = \boldsymbol{I}$, while 1 denotes the unit vector and $\boldsymbol{I}$ the unit matrix. Note that the matrices P of this section are not the same as those of Section 2. Application of the weighted residual method gives the system

$$\hat{A}\hat{\boldsymbol{c}} = \hat{\boldsymbol{b}}, \tag{5.1}$$

of order $K + K'$, or in more detail

$$\sum_{k=1}^{K} \langle \boldsymbol{w}_l, A P_{(\mathrm{mult})_k} \boldsymbol{x}^{(t)} \rangle c_{(\mathrm{mult})_k} + \sum_{k'=1}^{K'} \langle \boldsymbol{w}_l, A P_{(\mathrm{add})_{k'}} 1 \rangle c_{(\mathrm{add})_{k'}}$$
$$= \langle \boldsymbol{w}_l, \boldsymbol{b} \rangle - \langle \boldsymbol{w}_l, A \sum_{k'=1}^{K'} P_{(\mathrm{add})_{k'}} \boldsymbol{x}^{(t)} \rangle, \tag{5.2}$$

Table 2
Cost for multiplicative and hybrid correction of trial problem

Type	Number of processors	Global iterations	Plane passes
Hybrid	2	7	2312
Multiplicative	2	9	2310
Hybrid	4	16	2674
Multiplicative	4	15	3055
Hybrid	8	18	3614
Multiplicative	8	21	3446

for $l = 1, 2, \ldots, K + K'$. Galerkin weighting is used. Examination of the first term on the left of (5.2) shows those rows of the matrix A corresponding to coarse mesh grid points are mostly those of the corresponding rows in the multiplicative form (2.6). Similarly it is concluded the rows of A corresponding to fine mesh grid points are largely those of the corresponding rows on the additive form (2.7). The only differences in each case occur for some elements of A at interface grid points between the two schemes. Examination of the left-hand side of (5.2) reveals the matrix $\hat{A}$ is symmetric. It can easily be shown to be positive definite. Provided $\boldsymbol{e}^{\mathrm{T}} \neq \mathbf{0}$ and $\boldsymbol{x}^{(l)} > \mathbf{0}$:

$$\boldsymbol{e}^{\mathrm{T}}\hat{A}\boldsymbol{e} = \left(\sum_{k=1}^{K} e_k P_{(\mathrm{mult})_k} \boldsymbol{x}^{(l)} \right)^{\mathrm{T}} A \left(\sum_{k=1}^{K} e_k P_{(\mathrm{mult})_k} \boldsymbol{x}^{(l)} \right)$$

$$+ \left(\sum_{k'=1}^{K'} e_{k'} P_{(\mathrm{add})_{k'}} 1 \right)^{\mathrm{T}} A \left(\sum_{k'=1}^{K'} e_{k'} P_{(\mathrm{add})_{k'}} 1 \right). \tag{5.3}$$

It follows (5.3) is positive definite because $\boldsymbol{a}^{\mathrm{T}} A \boldsymbol{a} > 0$ for any $\boldsymbol{a} \neq \mathbf{0}$. Consequently the intermediate systems (5.1) may be solved by the preconditioned conjugate gradient scheme.

The same trial problem was then evaluated for the hybrid technique and the difficulties that arise in the early stages of the iteration due to additive correction vanish. The results for the hybrid and multiplicative forms are shown in Table 2, and the tight convergence used in Section 4 is not employed [2]. The number of global iterations is shown and a rough approximation to the computational effort is recorded. This measure is the number of preconditioned conjugate gradient passes through each (x, y)-plane of the three-dimensional structure. The results for the hybrid and multiplicative technique involve comparable effort on both measures. The cost for additive corrections (not shown) is considerably greater.

6. Conclusion

The simple form of additive correction first tested here is not effective for the parallel algorithm based on CMR. The hybrid form overcomes difficulties associated with it, and is as effective in iteration counts as the multiplicative form, but is slightly less costly to implement.

References

[1] J.M. Barry and J.P. Pollard, A method of implicit non-stationary iteration for solving neutron diffusion linear systems, *Ann. Nuclear Energy* **4** (1977) 485–493.
[2] J.M. Barry, J.P. Pollard and E.L. Wachspress, Some parallel iterative schemes for large systems of linear equations arising in three-dimensional modelling, to appear.
[3] A. Brandt, Multi level adaptive solutions to boundary-value problems, *Math. Comp.* **31** (1977) 333–390.
[4] G.H. Golub and C.F. Van Loan, *Matrix Computations* (Johns Hopkins, Baltimore, MD, 1983).
[5] J.A. Meijerink and H.A. van der Vorst, An iterative solution method for linear systems of which the coefficient matrix is a symmetric M-matrix, *Math. Comp.* **31** (1977) 148–162.
[6] S. Nakamura, *Computational Methods in Engineering and Science with Applications to Fluid Dynamics and Nuclear Systems* (Wiley, New York, 1977).
[7] E.L. Wachspress, *Iterative Solution of Elliptic Systems and Applications to the Neutron Diffusion Equation of Reactor Physics* (Prentice-Hall, Englewood Cliffs, NJ, 1966).